Higher isn't Necessarily Better: Visibility Algorithms and Experiments

Wm Randolph Franklin
Rensselær Polytechnic Institute
Troy, NY, 12180 USA

Phone: +1 (518) 276–6077
Fax: +1 (518) 276–6261
Internet: wrf@ecse.rpi.edu

Clark K Ray
Department of Electrical Engineering and Computer Science
U.S. Military Academy
West Point, NY 10996 USA
Internet: dc6119@eecs1.eecs.usma.edu

May 28, 1994

Abstract

We describe several fast programs to compute viewsheds and weighted visibility indices for observation points in a raster terrain. These programs explore various tradeoffs between speed and accuracy. We have analyzed many cells of data; there is no strong correlation between a point's elevation and its weighted visibility index. However, the, very few, high visibility points tend to characterize features of the terrain. This work can form a basis for automatically locating observers jointly to cover a terrain region of interest.

Keywords: visibility, viewshed, line-of-sight.

1 Introduction

Visibility research, calculating lines-of-sight and viewsheds, on terrain databases, is an established, and important, field in GIS. Nevertheless, progress is still possible, since both more powerful Unix workstation hardware, with tens of megabytes of memory and software environments, and

much larger amounts of input data are available. The research reported here would not have been feasible on mainframes nor possible on personal computers. Ninety meter resolution coverage of the US is available from the USGS, and of other regions for appropriate purposes from DOD, in the Digital Elevation Model (DEM), or equivalently, the Digital Terrain Elevation Data (DTED) format. Higher resolutions can be available for specific purposes. Synthetic aperture radar can produce elevation data at a 1 meter resolution; about the biggest problem here is how to store it. Elevation data of other planets, such as Mars, is available.

Some notable earlier work is as follows. Wood and Fisher, 1993 consider the problem of elevation data accuracy. Fisher, 1992; Fisher, 1993 discusses viewshed uncertainty, while Lee et al., 1993 describe what data errors do to feature extraction. Lee, 1992 considers how high visibility points relate to topographic features. For a definitive survey listing older references, see Nagy, 1994. De Floriani et al., 1986 give some of the relevant issues in using Triangulated Irregular Network data structures, instead of the raster databases used here.

The long-term goal of this work is to facilitate development of an automated procedure to place facilities, given information on terrain and the area to be defended for the US Army Topographic Engineering Center (formerly called the Engineer Topographic Labs), part of the Corps of Engineers, Ft. Belvoir, Virginia. Army engineers need to know where to place facilities, or alternatively, where to hide. In fact, students at West Point are taught how to calculate this approximately by hand on a contour map. Some techniques are described in Shannon and Ignizio, 1971.

If this is used to locate facilities, then, after calculation, then the fraction of the potential viewshed which is visible from the observer, or the visibility index, may be postprocessed to select points that also have other desirable properties, e.g., that are accessible on the ground.

However, many other applications are possible for this work.

1. Cellular telephone companies, and other radio transmitter operators, wish to optimize their antennas' locations. This application also illustrates the utility of having the observer and target heights being different. Eventually, these Line of Sight algorithms might be modified to account for attenuation effects as described in Conrad, 1993.

2. Natural resource extractors may wish to site visual nuisances, such as clearcut forests and openpit mines, where they cannot be seen from public roads. Also, zoning laws in some regions, such as the Adirondack Park of New York State, may prohibit new buildings that can be seen from a public lake.

3. Finally, if other planets, such as Mars, are being explored by semi-autonomous roving vehicles communicating with a fixed ground base,

then we need to know where to put that base, and where it is safe to rove around it. Gennery, 1989 describes matching a local terrain description obtained from a Mars rover with a more global terrain description from an orbiting probe. Since the terrain points of higher visibility help to characterize a surface, matching these particular points would make the correlation-based approach even more effective.

The short term goal of this work is extracting key visibility information from the mass of terrain elevation data in a concise form that would facilitate the use of any siting model, that is, calculating a viewshed.

We are attacking a narrow technical problem here, and ignore many important issues that must be considered in an actual useful system. This includes data reliability, the effect of vegetation, light refraction, etc. Our strategy is to study one facet deeply first. Nevertheless, this does affect other facets, e.g., we have observed that points with the highest visibility seem to be least affected by uncorrelated errors in the elevation data.

Much of the work reported here is from Ray, 1994. This is a preliminary paper; some conclusions might be modified as we proceed.

2 Technical Details

There are several necessary definitions and other preliminaries. The lines-of-sight (LOSs) radiate away from the observer, who is some fixed distance above ground level such as 5 meters. The observer is trying to see the target, also at a fixed distance above the ground, such as 25 m. If the observer height is different from the target height, then visibility is not reflexive. That an observer above point A on the ground can see a target above point B does not necessarily mean that an observer above B can see a target above A. We are interested only in targets within a certain fixed range, or radius of interest, of the observer, such as 40 km.

We have no reason to suspect that the particular values of the above parameters, which are useful in aircraft radar tracking, affect the nature of the results, at least within wide ranges, so we generally use the stated values. Note that our range, about 500 pixels, is larger than often reported in the literature, about 100–200 pixels.

The curvature of the earth can be most easily compensated for by systematically changing elevations before processing, at least for regions much smaller than the earth's radius.

We work with raster elevation databases, such as DTED or DEM, which has 1201x1201 elevations for a 1 degree times 1 degree region, such as the following datasets.

1. Korean DTED data, such as cell number N37E127. This has a nice mix of a low-lying valley with a mountain range. See figure 1, which shows

the NW quadrant of that cell.

2. The Lake Champlain West DEM file with a section of the Adirondack mountains in New York State.

3 Programs Written

We have written several programs so far, which we'll now describe.

3.1 DEM/Xdraw

DEM/Xdraw calculates the viewshed region in an elevation database as seen by a given observer, by determining, for each point in the database, whether a target at that point would be visible. The very fast, but approximate, method grows a ring of LOS information out from the observer. Consider the one-dimensional case first, as shown in Figure 2(a). The observer is at ground level at 0. Point 1 is always visible. We run a LOS from the observer through 1 and see that it falls below point 2. Therefore 2 is visible, and we raise the LOS to go through 2, as shown, dotted.

This LOS goes below 3, so 3 is visible, and we calculate a new LOS through 3, shown, dashed. Points 4 and 5 are below this, so the are hidden. 6 is above, so 6 is visible, and we raise the LOS, shown thin solid. 7 is below, and so is hidden. Two things are calculated for each point: whether the point is visible, and if it is hidden, the elevation of a LOS from the observer passing over it.

Note that determining the visibility of point #i requires information about only point #(i-1); we do not check any points closer to the observer. This obvious information is presented to set the stage for the 2-D case. Here we calculate LOSs and visibility for a square ring of points around the observer at a time, as shown in figure 2(b). (This is similar to the parallel algorithm in Teng and Davis, 1992.) When processing a point, we look only at points in the next ring in, and if a point is hidden, we determine the elevation of the LOS above it. If a point is in direct line with a point in the next inner ring, then this reduces to a 1-D problem. However, this is true for only the points along the eight principal directions from the observer.

For the other points, we interpolate a line of sight. For example, consider point (2,1). The LOS from it to the observer at the origin, goes between (1,0), and (1,1), so we calculate an approximate LOS from the observer to (2,1) from their elevations. There are various obvious possibilities: to use the maximum of their elevations, the minimum, or to linearly interpolate between their elevations.

The maximum and minimum are chosen to be conservative in one direction or the other. With the maximum, LOSs will probably be calculated higher than they should be. Thus, any point that is found to be visible will almost

certainly really be visible. With the minimum, any point found to be hidden will almost certainly be hidden. Linear interpolation should be more accurate, but is slower. By processing a database three times, with linear interpolation, maximum, and minimum, we obtain a good estimate for the viewshed, with error bars.

How we handle ring 2 is rather traditional; the change comes with ring 3. Consider point (3,1), whose LOS goes between (2,0) and (2,1). When finding the LOS for (3,1), we use, not the actual elevations of (2,0) and (2,1), but the elevations of their LOSs, which are higher if those points are hidden. This information contains information about all points adjacent to (3,1)'s LOS that are closer to the observer, so we do not check any other points, to determine (3,1)'s LOS. In summary, to process a point, we use information at only two other points, which is very fast. R2 is within a constant factor as fast and highly accurate, but potentially much more complex to implement robustly.

The problem with DEM/Xdraw is that the calculated visibility of a point may depend on points which are more than distance one away from the line to the observer. Consider (5,2). It depends on (4,1) and (4,2). They depend on (3,0), (3,1), and (3,2), which depend on (2,0), (2,1), and (2,2). (2,2) is not adjacent to the line from (0,0) to (5,2). The spread of involved points broadens the farther the target is from the observer. However, with linear interpolation, the influence of the more offline points is small, unless they contain a very dominant feature.

The way that DEM/Xdraw approximates contrasts to other methods, which fire a small number of rays from the observer. DEM/Xdraw uses each point too much, while the others ignore most points. Which is better needs to be determined experimentally. An advantage of DEM/Xdraw is that it this tends to smooth the surface a little, which may be good.

We implemented DEM/Xdraw as a C program on a Sparcstation 10/30. A typical execution time is 3.5 CPU seconds to calculate the visibility with the interpolation rule and display a 600x600 image, and 15 CPU seconds for a 1201x1201 image. To illustrate the machine's general speed, merely reading that image as a binary file uses 1.8 CPU seconds.

With these speeds, it is questionable whether there is a need for a parallel implementation, unless real-time output is desired as the observer moves. In this case, it might be better to have each frame calculated by a separate processor in parallel, instead of having the processors share the calculation of each frame.

Figure 3 shows a 600 by 600 extract from the Adirondack DEM, while Figure 4 shows approximate viewsheds for the same region using the max, min, and interpolation methods. The observer is 20 meters above the highest point, which is near the lower left, and the possible targets are 20 above the ground. The figures' intensities were histogram-equalized to improve legibility. In Figure 4, the white region is visible by the max approximation, and so is almost certainly visible. The light grey region is hidden by the max

approximation, but is visible by the interpolation method, and so is probably visible. The dark grey region is according to the interpolation method, but visible by the min method, and so is possibly visible, while the black region is hidden even by the min method, and so is almost certainly hidden.

3.2 R3

R3 calculates the viewshed around an given observer in the traditional method. It runs a separate LOS from the observer to each point within the circle, and sees whether any elevations on that LOS obstruct it. If the range is R, then the execution time is proportional to R cubed. R3 can handle user-specified regions of interest; they do not need to be circles around the observer.

Proc-LOS was used for massive visibility index calculations on about 20 DTED cells on the Korean peninsula.

3.3 R2

R2 is an optimization of R3, that proceeds as follows.

1. For each point on the perimeter of the range, calculate whether that point is visible by running a LOS from the observer to it, and checking the elevation of every point adjacent to the LOS, to see whether any will hide it.

2. Then work along each line of sight, processing the points adjacent to it as if this were a one-dimensional problem as described above, to determine which points are visible. A particular point may receive several visibility estimates in a series as a sequence of LOS's "sweep" past it; it retains the estimate from the most spatial proximate LOS.

If the range is R, then R2 takes times proportional to R squared. The results from R2 are close to R3. In one test, we tried both methods from the same observer point on the SW quadrant of cell N37E127. The height of the LOS above each point, as calculated by each method differed by up to 67 meters, but the extreme was rare. The mean difference was -0.82 m, with the standard deviation 3.4 m. Also, of the 276,460 points in the region of interest, the algorithms agreed on the visibility of 273,144, or 98.8%. In return for this loss of accuracy, the execution time sped up from 896 CPU seconds to 18.

3.4 LOS

This set of programs calculates a weighted visibility index for each point of the database. That is, how much of the area of a circle of a certain radius can an observer placed at each point in turn see? The approach used here implicitly

weights the visibility of areas close to the observer more than areas further away.

LOS is a fast approximate line-of-sight (LOS) program intended for eventual migration to a parallel machine. It is under 1000 lines of C code. LOS calculates a visibility index for each point in an elevation grid. To calculate the visibility index for a point, LOS fires a small number of rays, such as 16, from the point out to the radius of interest. The validity of using only a small number of rays was demonstrated experimentally by Ray, 1994. A theoretical justification can be given as follows.

The possible target points within the radius of interest for this given observer form a set, S. Each point is either hidden or visible. The visibility index is the average number of visible points, that is, is a statistic of S, and by the law of large numbers a mean of a set can be determined by sampling a small number of observations. Since $|S|$ is finite, problems such as infinite population variances, which would cause the sample means not to converge, do not occur. The only requirement is that whether a point is sampled must not be correlated with its value. This might happen if there were strong vertical features that ran exactly along lines of constant latitude or longitude. However, this has been observed only for clearly erroneous data.

Therefore we are comfortable with only certain angles being sampled. The next step is to sample only certain points in each direction, that is, when we get far enough out from the observer, we don't use every pixel. Every time we process another 16 pixels in this direction, we double the distance between pixels used. From one to 16 pixels away, we use every pixel. From 17 to 48 pixels away, we use every second pixel. From 49 to 112 pixels away from the observer, we use every fourth pixel, etc. This rule was chosen for its speed of execution. Asymptotically, if the line of sight has N pixels, then we use proportional to log(N) of them.

We devoted considerable effort to optimizing LOS. This was done specifically on a Sun IPC, though the techniques are generally applicable. They include using integers instead of floating numbers, comparing various compilers, and creating (with a preprocessor) a separate block of code for each different slope of the line of sight.

On a Sun IPC workstation, the execution time ranged from 24 CPU seconds for a 300x300 scene with a range of 50, up to 568 CPU seconds for a 1201x1201 scene with range of 100. As expected, the time varies linearly with the number of points examined.

3.4.1 Feature Detection in Images

The ability to enhance features, such as edges, in a photographic image is an unexpected property of LOS. We have done preliminary studies in this, and offer it as as area for future research. The idea is to consider each pixel's intensity as if it were elevation when calculating visibility indices. This can

enhance even edges bordered by gradual intensity changes.

3.5 Proc-LOS

Proc-LOS calculates a weighted visibility index for each of the 1,442,401 possible observers in a cell. The method is as follows. For each point in the cell, place an observer there and fire a number of rays out to the range to determine how many points on the ray are visible. This method is traditional, but the amount of data processed is new. To date, about 20 cells have been processed.

How many rays are necessary? At first, we used 128 rays per point, which required 1,454,000 CPU seconds on a Sun IPC per cell. (A Sun IPC is 3—4 times slower than the newer Sun 10/30 used for many of the other experiments reported here.) If 32 rays were sufficient, then that would cut the CPU time down to 357,000 seconds per cell. (By now, these times have been considerably improved.) The sufficiency of 32 rays was shown two ways. First, the visibility index of each point in a cell was calculated with 128 rays, and again with 32 rays, and the two visibility indices correlated. Figure 5 plots a random sample of the visibility indices calculated with the two methods against each other. There's almost no difference.

However, the reason for finding visibility indices is to find the best points and place observers there. Even though on average the two methods are equivalent, they might differ on the best points. Therefore we selected the dozen points of highest visibility with the each method, and compared them, as shown in figure 6. Ten of the 12 points are common to both lists, and most points are in the same order. Therefore, 32 rays are quite sufficient.

Figure 6 shows another interesting point. The best points are well scattered around the cell, and not all adjacent to each other. This is good since it may not be necessary artificially to separate the points to get a good spread.

When the cell shown in figure 1 was processed, figure 7 resulted. There are some noteworthy observations here.

1. Visibility does not appear strongly correlated with elevation. On the one hand, the ridge in the middle is both high and visible. However, the low region in the bottom left is low and visible, while the high region near the top right is high but invisible. An explanation is that from a low, broad valley one can see the whole valley and the fronts of two ranges, while from a peak in a mountainous region one can see only the adjacent small valleys in front of the adjacent peaks.

2. The visibility index seems to sharpen features. The highly visible part of the central ridge is much narrower than the ridge itself. This method might be useful for identifying ridges and for feature recognition.

3. There are very few highly visible points. Figure 8 shows a semilog graph of the distribution of points of different visibility indices. The visibility

index is scaled so that 32K means that 100% of the points within range are visible, 8K means 25%, etc. Of the 1.4 million possible observers, only about 40 can see over 1/2 of the possible targets within range. Therefore, selecting the best observers is difficult and critical.

4. The distribution of visibility indices is log-linear, like many natural statistics.

5. On the average, higher points are not much more visible. Figure 9 is a scatter plot of visibility index versus elevation for a random sample of points. There is no apparent correlation. In fact, the correlation here is -0.12. This agrees with experience, where the shoulder of a ridge may be more visible than the top.

6. Nevertheless, the best points are somewhat higher, although they are nowhere near the highest. Figure 10 shows the 100 best points' visibility and elevation plotted. Within this set, there is still no correlation between elevation and visibility. However the set has no points below 100 meters. On the other hand the set also has no points above 800 meters, even though the highest point is 1462 m.

3.5.1 Dependence of Visibility on Elevation

Whether or not the visibility index of a point is linearly correlated with its elevation, and whether that correlation is positive, depends on many factors, in a way that we don't yet understand. The factors include which cell is tested, how much ocean that cell contains, whether we count the sea-level points, whether we weight points either closer to, or farther from, the observer more highly, whether we use all the points, or only the best points, etc, etc. The most that we can say is that the correlation, on the average, is weakly positive in some cases, with a very large standard deviation.

Nevertheless, simply placing observers on the highest points is in many cases not the optimal method of covering some terrain.

3.5.2 Directional Visibility

We have extended Proc-LOS to calculate the visibility index of each point in each of several directions, and then to display the data in color. The hue of each point tells in which direction an observer could see the best. The color saturation tells how directional the visibility is, while the brightness still tells how good the overall visibility is here.

The directional visibility is quite good at highlighting terrain features such as ridges.

3.6 Demonstration Program to the User Community

Although the generation of a visibility grid corresponding to the elevation data for an area of interest provides the basis for extracting concise information for a variety of siting algorithms, it also offers the opportunity to provide a display to augment existing ways of visually representing terrain considerations for planners and decision makers. Many users of digital terrain data who could provide useful insights into various methods of rendering a visibility image may not have access workstation environments common in research and academic settings.

In an attempt to make some form of visibility display available to users with relatively restricted computing resources, a standalone demonstration system was developed whose primary screen display is shown in Figure 11. It can execute on any Intel 80x86 architecture running an MS-DOS or compatible operating system with a standard VGA (640x480x16 color) display. The entire demonstration package is distributed on a single 1.2MB 5.25 inch floppy disk, which can hold the demonstration program, background and operating information text, and elevation and visibility data for 12 areas corresponding to one-degree DTED Level I cells. Also provided for each area are lists of the 16 best and 64 best visible points, as selected from the entire cell and as selected from within 16 equal sub regions with a cell to reduce clustering.

This demo has been quite well received.

4 Conclusions

Various speed–accuracy tradeoffs were made in the viewshed programs. With the very fast, but less accurate, Xdraw program, we calculated a fuzzy visibility for each point. R3, the slowest program, was most accuracte, while R2 was much faster and almost as good. In the visibility index programs, LOS calculated a very fast approximate weighted index for each possible observer. Proc-LOS tested how many rays needed to be fired from each observer, and founds that 32 was as good as 128, both for the mass visibility indices, and for identifying the best points. Proc-LOS was run on 30,000,000 points. Some of its results were encapsulated in a PC-based demo program to help in observer siting.

Massive visibility index and viewshed calculations on databases have produced some tantalizing results. The question is now whether they extend to other DEMs, to data at higher resolutions, and to nonterrestrial data, such as Mars and Venus. Also, since the available data has frequent errors, and even when correct, it's not always obvious what the elevation of the representative point in each 3"x3" rectangle means, further investigation into the robustness of these results is indicated. We are now continuing this research and extending it automatically to site observers to cover some specified terrain.

5 Acknowledgements

This work was supported by NSF grant CCR-9102553. Partial support for this work was provided by the Directorate for Computer and Information Science and Engineering, NSF Grant No. CDA-8805910. We also used equipment at the Computer Science Department and Rensselær Design Research Center at RPI. Financial support for research to date has been provided by the U.S. Army Topographic Engineering Center, Ft. Belvoir, Virginia. Some equipment and software have been provided by the Department of Electrical Engineering and Computer Science, U.S. Military Academy, West Point, New York. Some work was done at the Division of Information Technology, Commonwealth Scientific and Industrial Research Organization, Canberra, Australia.

John Bradley's excellent image processing program XV was used for much of the image postprocessing and display. Jef Poskanzer's Portable BitMap (PBM) suite of conversion utilities was also valuable.

The views, opinions, and/or other findings contained in this report are those of the authors and should not be construed as an official National Science Foundation or Department of the Army position, policy, or decision, unless so designated by other documentation.

References

Conrad, A. (1993). Software tracks terrain effects on transmitters. *Microwaves & RF*, 32:141. Product evaluation.

De Floriani, L., Falcidieno, B., Pienovi, C., Allen, D., and Nagy, G. (1986). A visibility-based model for terrain features. In *Proceedings: Second International Symposium on Spatial Data Handling*, pages 235–250. Seattle, Washington.

Fisher, P. F. (1992). 1st experiments in viewshed uncertainty - simulating fuzzy viewsheds. *Photogrammetric Engineering And Remote Sensing*, 58:345–352.

Fisher, P. F. (1993). Algorithm and implementation uncertainty in viewshed analysis. *International Journal Of Geographical Information Systems*, 7:331–347.

Gennery, D. B. (1989). Visual terrain matching for a Mars rover. In *IEEE Conf. on Vision and Pattern Recognition*, pages 483–491, San Diego, CA, USA. RN has.

Lee, J. (1992). Visibility dominance and topographic features on digital elevation models. In Bresnahan, P., Corwin, E., and Cowen, D., editors,

Proceedings 5th International Symposium on Spatial Data Handling, volume 2, pages 622–631. International Geographical Union, Commission on GIS, Humanities and Social Sciences Computing Lab, U. South Carolina, Columbia, South Carolina, USA.

Lee, J., Snyder, P. K., and Fisher, P. F. (1993). Modeling the effect of data errors on feature extraction from digital elevation models. *Photogrammetric Engineering And Remote Sensing*, 58:1461–1467.

Nagy, G. (1994). Terrain visibility. Technical report, Computational Geometry Lab, ECSE Dept, Rensselaer Polytechnic Institute.

Ray, C. K. (1994). *Representing Visibility for Siting Problems*. PhD thesis, Electrical, Computer, and Systems Engineering Dept., Rensselaer Polytechnic Institute.

Shannon, R. E. and Ignizio, J. P. (1971). Minimum altitude visibility diagram — MAVD. *Simulation*, pages 256–260.

Teng, Y. A. and Davis, L. S. (1992). Visibility analysis on digital terrain models and its parallel implementation. Technical Report CAR-TR-625, Center for Automation Research, University of Maryland, College Park, Maryland.

Wood, J. D. and Fisher, P. F. (1993). Assessing interpolation accuracy in elevation models. *IEEE Computer Graphics And Applications*, 13:48–56.

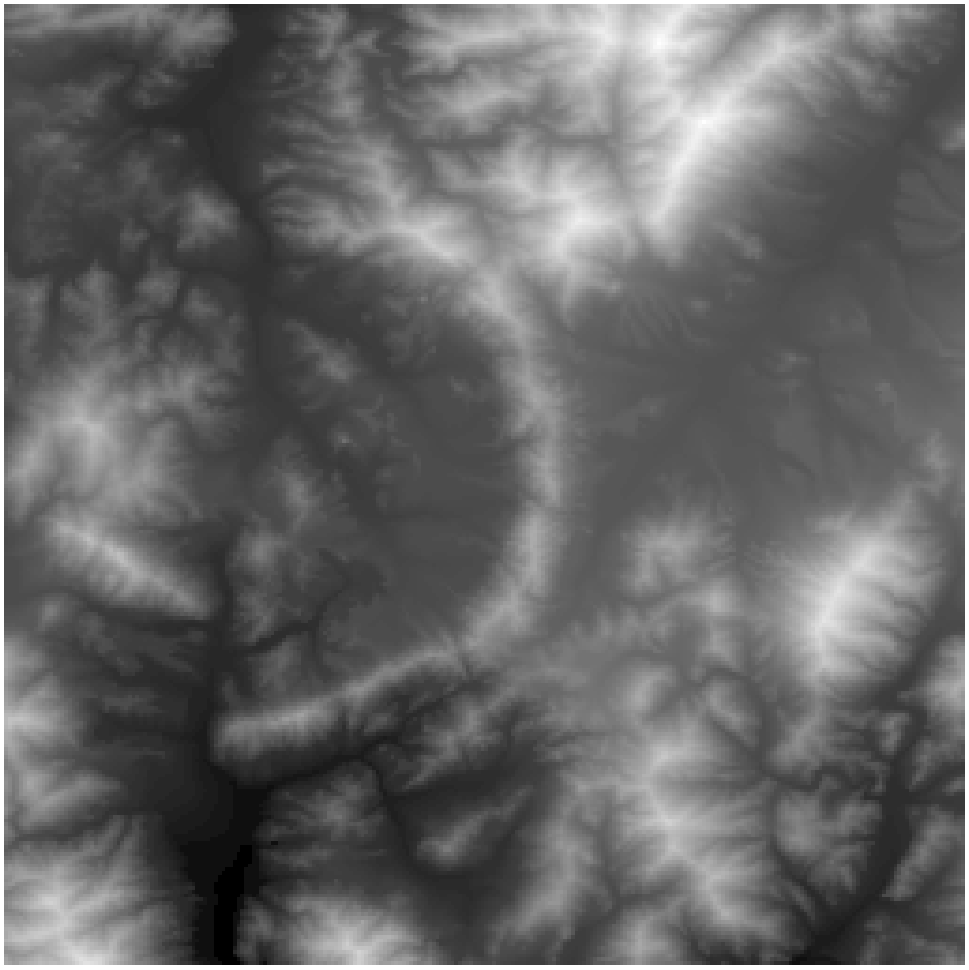

Figure 1: N37E127 DTED Cell

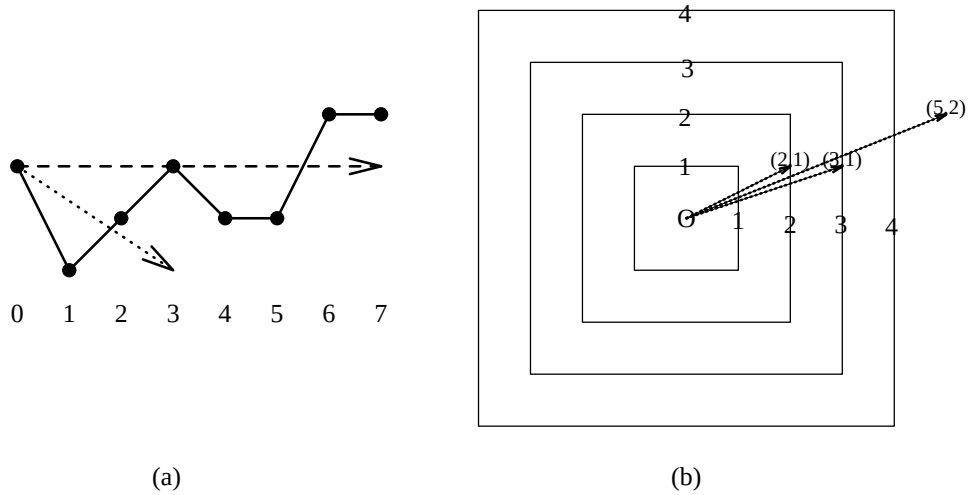

Figure 2: (a) One Dimensional Visibility. (b) Two Dimensional Visibility Ring

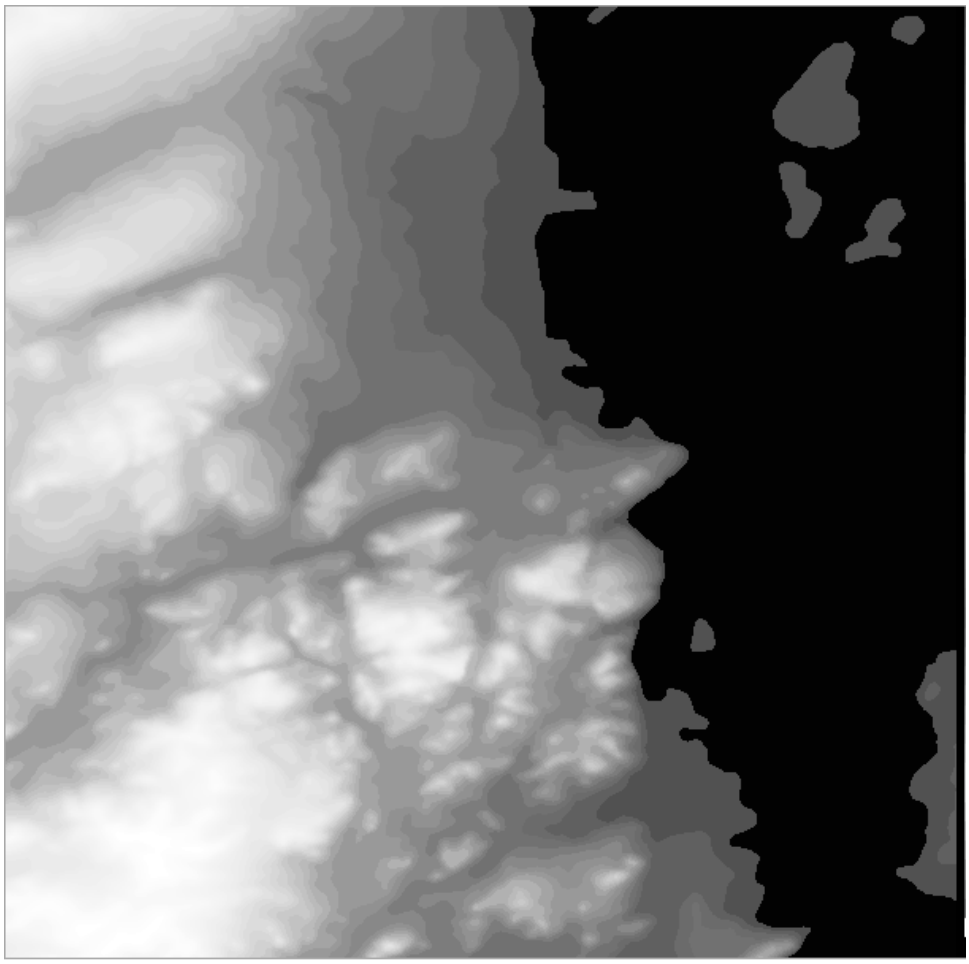

Figure 3: Sample Adirondack Terrain for DEM/Xdraw

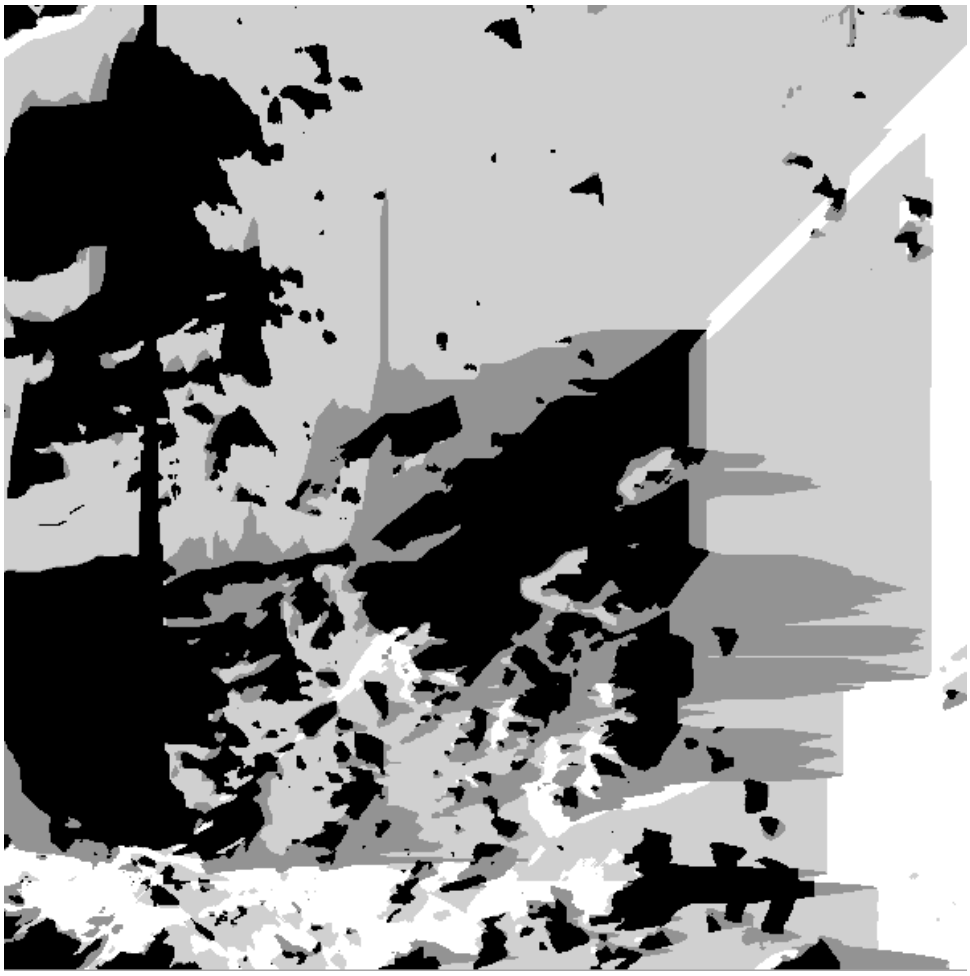

Figure 4: Three Different Approximate Viewsheds

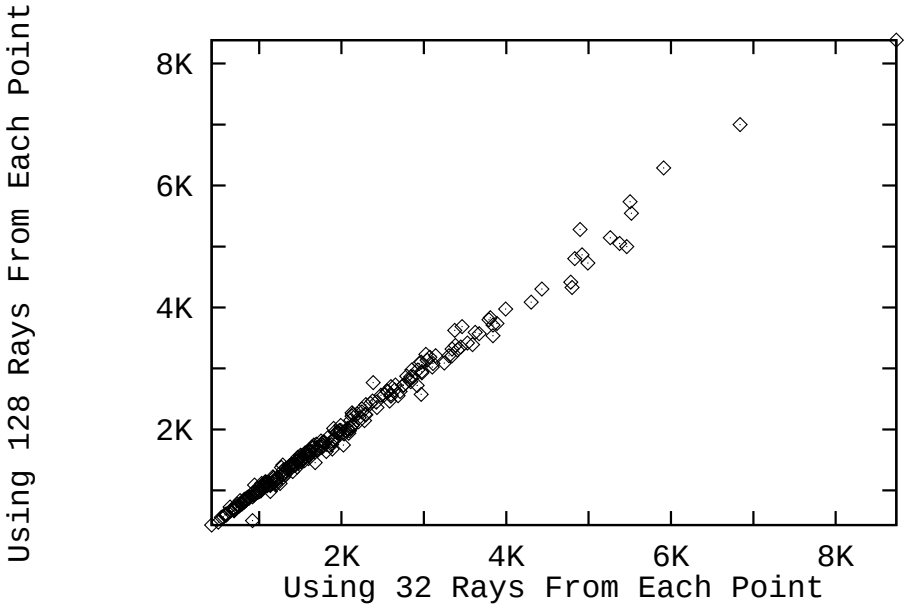

Figure 5: Comparison of Visibility Index Calculated with 32 vs 128 Rays per Point, for Some Random Points

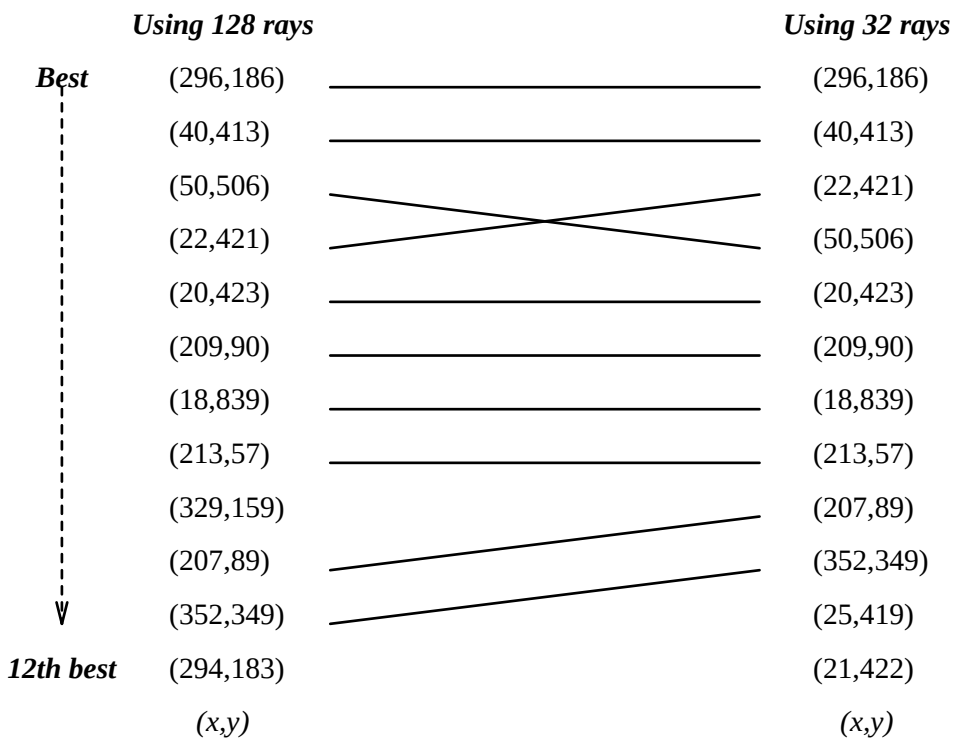

Figure 6: Best 12 Points, Selected Using 128 Rays vs 32 Rays

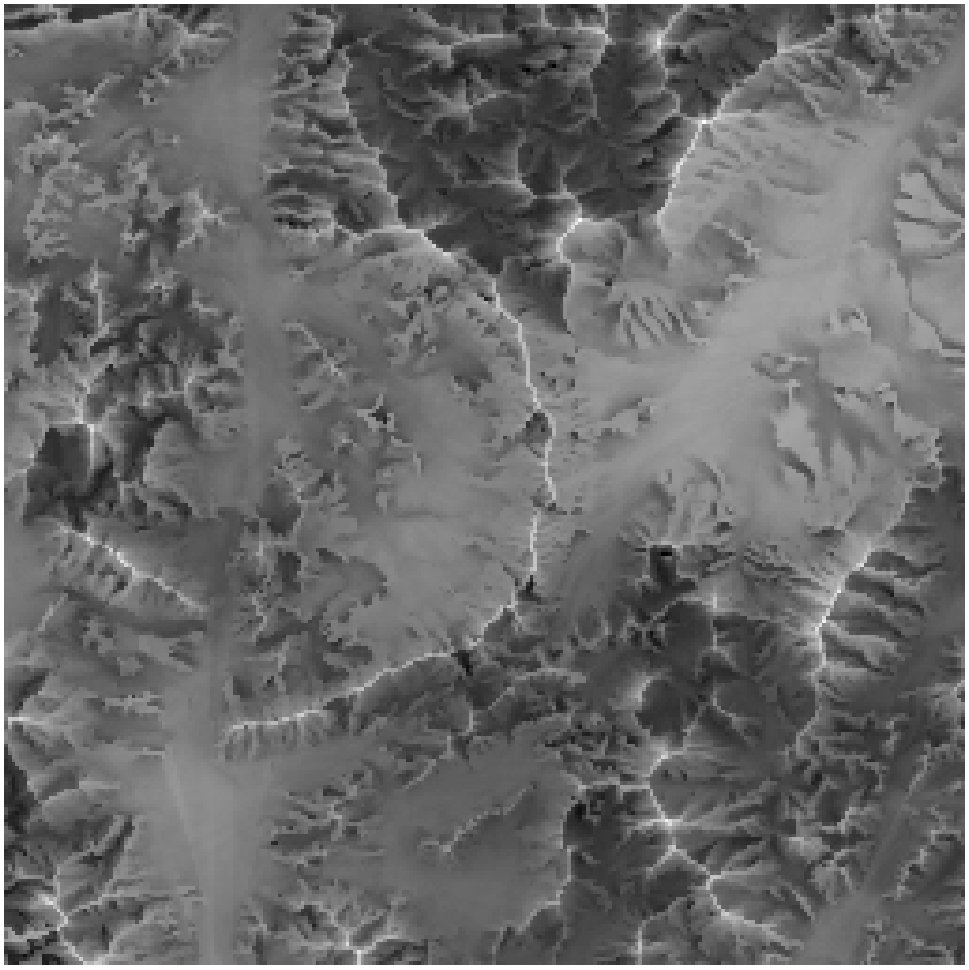

Figure 7: Visibility Index For Each Point

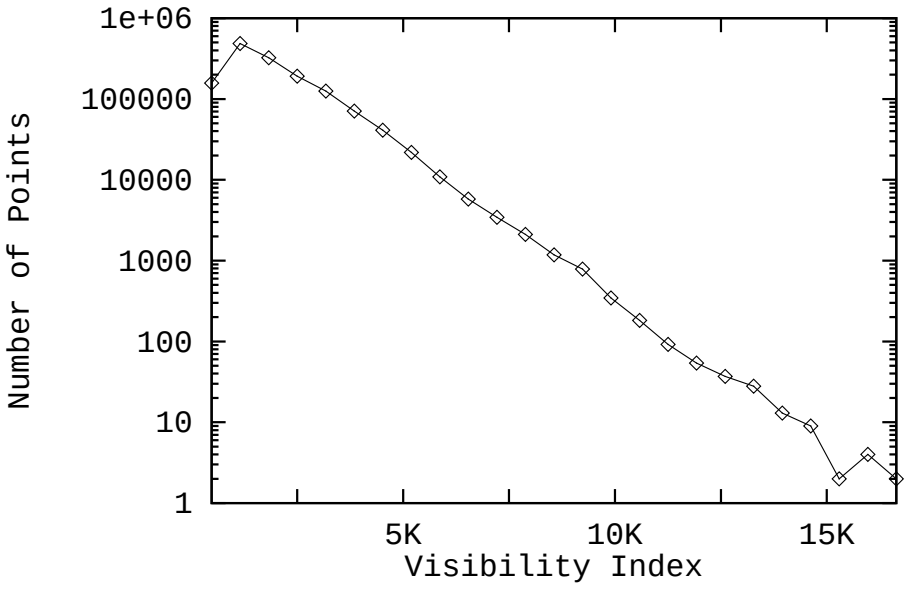

Figure 8: Histogram of Visibility Indices

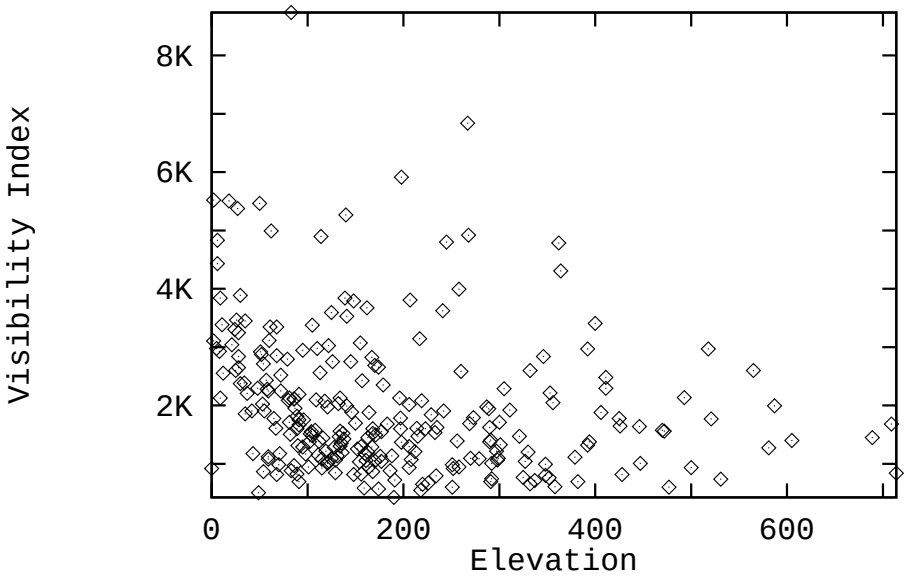

Figure 9: Visibility Index vs Elevation of Some Random Points

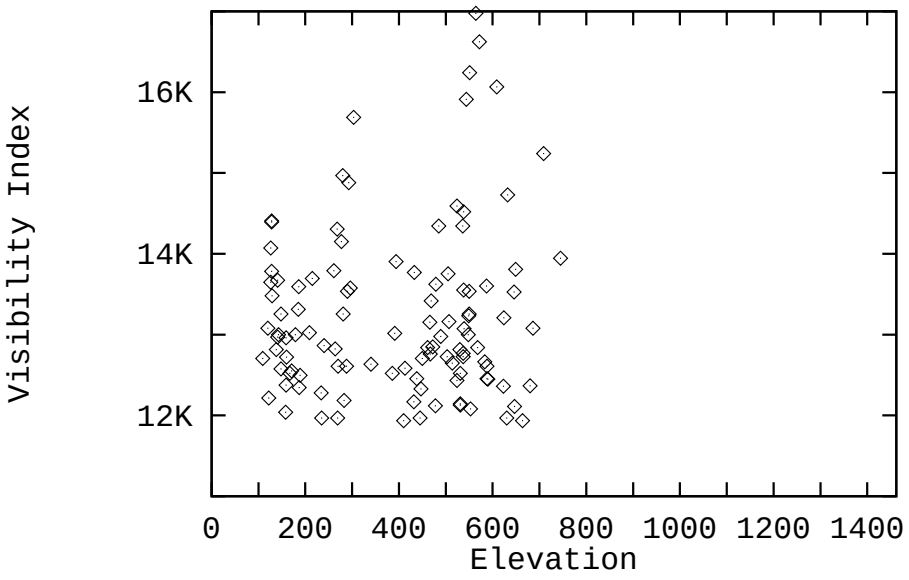

Figure 10: Visibility vs Elevation of the Best Points

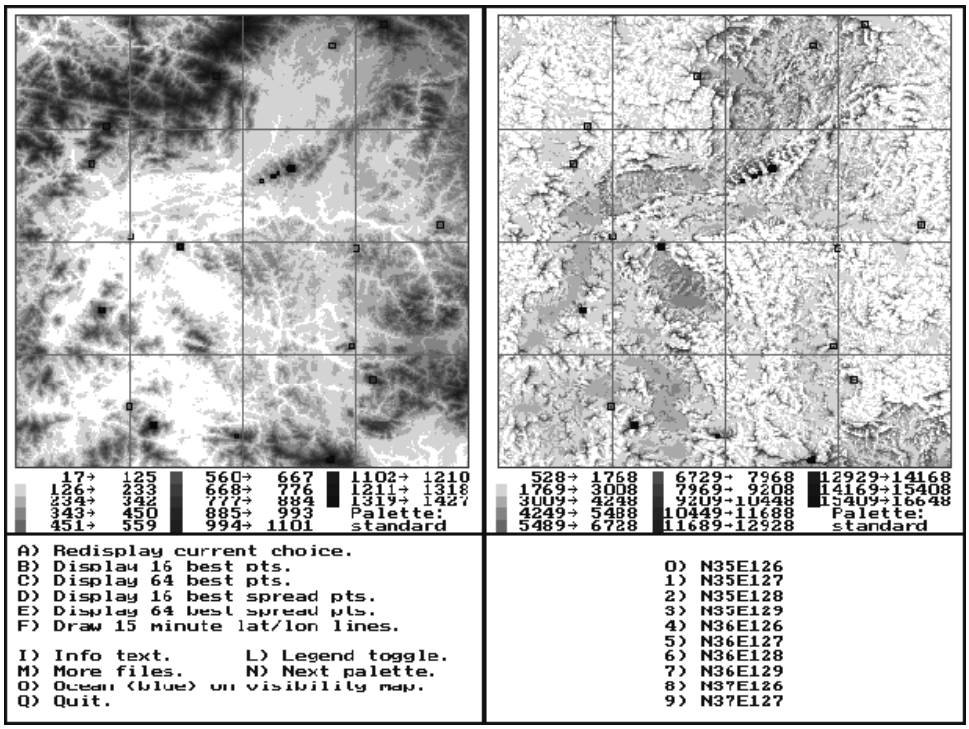

Figure 11: Screen from PC Demo Application